Freeform Search

Database:
- US Pre-Grant Publication Full-Text Database
- US Patents Full-Text Database
- US OCR Full-Text Database
- EPO Abstracts Database
- JPO Abstracts Database
- Derwent World Patents Index
- IBM Technical Disclosure Bulletins

Term: L8 NOT L9

Display: 50 Documents in Display Format: REV Starting with Number 1

Generate: ○ Hit List ◉ Hit Count ○ Side by Side ○ Image

Search Clear Interrupt

Search History

DATE: Saturday, May 12, 2007 Purge Queries Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
	DB=USPT; PLUR=NO; OP=OR		
L10	L8 NOT L9	89	L10
L9	L8 and (obfuscate or obfuscated or obfuscation).ab.	3	L9
L8	L7 or l6 or l5	92	L8
L7	L2 and (program adj counter)	91	L7
L6	L2 and (instruction adj counter)	1	L6
L5	L3 and (program adj counter)	12	L5
L4	L3 and (instruction adj counter)	0	L4
L3	L2 and ((smart ADJ card) or (java ADJ card))	59	L3
L2	obfuscate or obfuscated or obfuscation	1083	L2
L1	obsfuscation	0	L1

END OF SEARCH HISTORY

Term used **obfuscation** Found **584** of **201,062**

Sort results by relevance
Display results expanded form

Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 Software and systems: Obfuscation of design intent in object-oriented applications
Mikhail Sosonkin, Gleb Naumovich, Nasir Memon
October 2003 **Proceedings of the 3rd ACM workshop on Digital rights management DRM '03**
Publisher: ACM Press
Full text available: pdf(368.61 KB) Additional Information: full citation, abstract, references, index terms

Protection of digital data from unauthorized access is of paramount importance. In the past several years, much research has concentrated on protecting data from the standpoint of confidentiality, integrity and availability. Software is a form of data with unique properties and its protection poses unique challenges. First, software can be reverse engineered, which may result in stolen intellectual property. Second, software can be altered with the intent of performing operations this software m ...

Keywords: code generation, refactoring, software obfuscation

2 Privacy and anonymity: Obfuscated databases and group privacy
Arvind Narayanan, Vitaly Shmatikov
November 2005 **Proceedings of the 12th ACM conference on Computer and communications security CCS '05**
Publisher: ACM Press
Full text available: pdf(239.03 KB) Additional Information: full citation, abstract, references, index terms

We investigate whether it is possible to encrypt a database and then give it away in such a form that users can still access it, but only in a restricted way. In contrast to conventional privacy mechanisms that aim to prevent *any* access to individual records, we aim to restrict the set of queries that can be feasibly evaluated on the encrypted database.We start with a simple form of database obfuscation which makes database records indistinguishable from lookup functions. The only feasibl ...

Keywords: database privacy, obfuscation

3 Reliability and security: Hardware assisted control flow obfuscation for embedded processors
Xiaotong Zhuang, Tao Zhang, Hsien-Hsin S. Lee, Santosh Pande
September 2004 **Proceedings of the 2004 international conference on Compilers,**

architecture, and synthesis for embedded systems CASES '04
Publisher: ACM Press
Full text available: pdf(275.14 KB) Additional Information: full citation, abstract, references, citings, index terms

With more applications being deployed on embedded platforms, software protection becomes increasingly important. This problem is crucial on embedded systems like financial transaction terminals, pay-TV access-control decoders, where adversaries may easily gain full physical accesses to the systems and critical algorithms must be protected from being cracked. However, as this paper points out that protecting software with either encryption or obfuscation cannot completely preclude the control flo ...

Keywords: control flow graph, obfuscation

4 Short papers: Reasoning about obfuscated private information: who have lied and how to lie
Xiangdong An, Dawn Jutla, Nick Cercone
October 2006 **Proceedings of the 5th ACM workshop on Privacy in electronic society WPES '06**
Publisher: ACM Press
Full text available: pdf(104.12 KB) Additional Information: full citation, abstract, references, index terms

In ubiquitous environments, context sharing among agents should be made privacy-conscious. Privacy preferences are generally specified to govern the context exchanging among agents. Besides who has rights to see what information, a user's privacy preference could also designate who has rights to have what obfuscated information. By obfuscation, people could present their private information in a coarser granularity, or simply in a falsified manner, depending on the specific situations. Neverthel ...

Keywords: Bayesian networks, inference control, privacy protection, ubiquitous environments

5 Tool demonstrations II: LOCO: an interactive code (De)obfuscation tool
Matias Madou, Ludo Van Put, Koen De Bosschere
January 2006 **Proceedings of the 2006 ACM SIGPLAN symposium on Partial evaluation and semantics-based program manipulation PEPM '06**
Publisher: ACM Press
Full text available: pdf(567.92 KB) Additional Information: full citation, abstract, references

This paper presents LOCO, a graphical, interactive environment to experiment with code obfuscation and deobfuscation transformations, which can be applied automatically, semi-automatically and by hand. LOCO is an extension of the multi-platform visualization tool LANCET, combined with an obfuscation infrastructure in the underlying link-time program rewriter DIABLO. By use of LOCO, a developer can easily navigate through the control flow graph of a program and do fine-grained obfuscation, test n ...

Keywords: binary rewriting, code obfuscation, security

6 Session 11A: On obfuscating point functions
Hoeteck Wee
May 2005 **Proceedings of the thirty-seventh annual ACM symposium on Theory of computing STOC '05**
Publisher: ACM Press
Full text available: pdf(333.82 KB) Additional Information: full citation, abstract, references, citings, index

terms

We investigate the possibility of obfuscating point functions in the framework of Barak et al. from Crypto '01. A point function is a Boolean function that assumes the value 1 at exactly one point. Our main results are as follows:We provide a simple construction of efficient obfuscators for point functions for a slightly relaxed notion of obfuscation, for which obfuscating general circuits is nonetheless impossible. Our construction relies on the existence of a very strong one-way permutation, a ...

Keywords: obfuscation

7 A framework for obfuscated interpretation
Akito Monden, Antoine Monsifrot, Clark Thomborson
January 2004 **Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation - Volume 32 ACSW Frontiers '04**
Publisher: Australian Computer Society, Inc.
Full text available: pdf(357.56 KB) Additional Information: full citation, abstract, references, index terms

Software protection via obscurity is now considered fundamental for securing software systems. This paper proposes a framework for obfuscating the program interpretation instead of obfuscating the program itself. The obfuscated interpretation enables us to hide functionality of a given program *P* unless the interpretation being taken is revealed. The proposed framework employs a finite state machine (FSM) based interpreter to give the context-dependent semantics to each instruction in *P* ...

Keywords: *encryption, obfuscation, software protection*

8 Manufacturing opaque predicates in distributed systems for code obfuscation
Anirban Majumdar, Clark Thomborson
January 2006 **Proceedings of the 29th Australasian Computer Science Conference - Volume 48 ACSC '06**
Publisher: Australian Computer Society, Inc.
Full text available: pdf(332.32 KB) Additional Information: full citation, abstract, references, index terms

Code obfuscation is a relatively new technique of software protection and it works by deterring reverse engineering attempts by malicious users of software. The objective of obfuscation is to make the logic embedded in code incomprehensible to automated program analysis tools used by adversaries. Opaque predicates act as tool for obfuscating control flow logic embedded within code. In this position paper, we address the problem of control-flow code obfuscation of processes executing in distribut ...

Keywords: code obfuscation, distributed predicate detection, distributed systems security, mobile code protection, opaque predicates, software protection

9 Software issues: Control flow based obfuscation
Jun Ge, Soma Chaudhuri, Akhilesh Tyagi
November 2005 **Proceedings of the 5th ACM workshop on Digital rights management DRM '05**
Publisher: ACM Press
Full text available: pdf(316.58 KB) Additional Information: full citation, abstract, references, index terms

A software obfuscator is a program O to transform a source program *P* for protection against malicious reverse engineering. O should be *correct* (O(*P*) has same functionality with *P*), *resilient* (O(*P*) is resilient against attacks), and *effective* (O(*P*) is not too much

slower than *P*). In this paper we describe the design of an obfuscator which consists of two parts. The first part extracts the control flow information from the program and ...

Keywords: control flow, software obfuscation

10 [Workshop papers: On instrumenting obfuscated java bytecode with aspects]
Kung Chen, Ju-Bing Chen
May 2006 **Proceedings of the 2006 international workshop on Software engineering for secure systems SESS '06**
Publisher: ACM Press
Full text available: pdf(92.02 KB) Additional Information: full citation, abstract, references, citings, index terms

Code obfuscators are widely used tools for protecting commercial Java software. Advanced obfuscation techniques make de-compiled Java programs not re-compilable, thus greatly raising the barrier of instrumenting Java bytecode for malicious purpose. However, we have found that the aspect-oriented programming language AspectJ can be abused to overcome advanced code obfuscation and to modify obfuscated Java software effectively using its bytecode instrumentation mechanism. This paper describes such ...

Keywords: AspectJ, Java, aspect-oriented programming, code obfuscation, software protection

11 [Emerging applications: Obfuscation of executable code to improve resistance to static disassembly]
Cullen Linn, Saumya Debray
October 2003 **Proceedings of the 10th ACM conference on Computer and communications security CCS '03**
Publisher: ACM Press
Full text available: pdf(155.75 KB) Additional Information: full citation, abstract, references, citings, index terms

A great deal of software is distributed in the form of executable code. The ability to reverse engineer such executables can create opportunities for theft of intellectual property via software piracy, as well as security breaches by allowing attackers to discover vulnerabilities in an application. The process of reverse engineering an executable program typically begins with disassembly, which translates machine code to assembly code. This is then followed by various decompilation steps that ai ...

Keywords: code obfuscation, disassembly

12 [Defensive technology: Detection of injected, dynamically generated, and obfuscated malicious code]
Jesse C. Rabek, Roger I. Khazan, Scott M. Lewandowski, Robert K. Cunningham
October 2003 **Proceedings of the 2003 ACM workshop on Rapid malcode WORM '03**
Publisher: ACM Press
Full text available: pdf(240.68 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper presents DOME, a host-based technique for detecting several general classes of malicious code in software executables. DOME uses static analysis to identify the locations (virtual addresses) of system calls within the software executables, and then monitors the executables at runtime to verify that every observed system call is made from a location identified using static analysis. The power of this technique is that it is

simple, practical, applicable to real-world software, and high ...

Keywords: anomaly detection, code analysis, dynamic analysis, execution monitoring, intrusion detection, malicious code detection, static analysis, system calls

13 Fast abstract session: network security: A control flow obfuscation method to discourage malicious tampering of software codes
Y. L. Huang, F. S. Ho, H. Y. Tsai, H. M. Kao
March 2006 **Proceedings of the 2006 ACM Symposium on Information, computer and communications security ASIACCS '06**
Publisher: ACM Press
Full text available: pdf(31.20 KB) Additional Information: full citation, abstract

The paper presents a control flow obfuscation method to discourage reverse engineering and malicious tampering of software codes. Given the original source codes and desired obfuscation criteria, the proposed method works by decomposing the source codes into fragments and then applying various transforms to the code fragments. As the output of our method, the transformed fragments are re-assembled and obfuscated with the designated obfuscation criteria. Moreover, since only control flows are obf ...

Keywords: control flow obfuscation, intellectual property protection, reverse engineering, software obfuscation, tamper-resistance

14 A semantics-based approach to malware detection
Mila Dalla Preda, Mihai Christodorescu, Somesh Jha, Saumya Debray
January 2007 **ACM SIGPLAN Notices , Proceedings of the 34th annual ACM SIGPLAN-SIGACT symposium on Principles of programming languages POPL '07**, Volume 42 Issue 1
Publisher: ACM Press
Full text available: pdf(712.01 KB) Additional Information: full citation, abstract, references, index terms

Malware detection is a crucial aspect of software security. Current malware detectors work by checking for "signatures," which attempt to capture (syntactic) characteristics of the machine-level byte sequence of the malware. This reliance on a syntactic approach makes such detectors vulnerable to code obfuscations, increasingly used by malware writers, that alter syntactic properties of the malware byte sequence without significantly affecting their execution behavior.This paper takes the positi ...

Keywords: abstract interpretation, malware detection, obfuscation, trace semantics

15 Attacks and countermeasures: Diversify sensor nodes to improve resilience against node compromise
Abdulrahman Alarifi, Wenliang Du
October 2006 **Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks SASN '06**
Publisher: ACM Press
Full text available: pdf(767.29 KB) Additional Information: full citation, abstract, references, index terms

A great challenge in securing sensor networks is that sensor nodes can be physically compromised. Once a node is compromised, attackers can retrieve secret information (e.g. keys) from the node. In most of the key pre-distribution schemes, the compromise of secret information on one node can have substantial impact on other nodes because secrets are shared by more than one node in those schemes. Although tamper-resistant hardware can help protect those secrets, it is still impractical for sensor ...

Keywords: diversity, obfuscation, reverse engineering, wireless sensor networks

16 Testing malware detectors
Mihai Christodorescu, Somesh Jha
July 2004 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 2004 ACM SIGSOFT international symposium on Software testing and analysis ISSTA '04**, Volume 29 Issue 4
Publisher: ACM Press
Full text available: pdf(374.57 KB) Additional Information: full citation, abstract, references, index terms

In today's interconnected world, malware, such as worms and viruses, can cause havoc. A malware detector (commonly known as virus scanner) attempts to identify malware. In spite of the importance of malware detectors, there is a dearth of testing techniques for evaluating them. We present a technique based on program obfuscation for generating tests for malware detectors. Our technique is geared towards evaluating the resilience of malware detectors to various obfuscation transformations commonl ...

Keywords: adaptive testing, anti-virus, malware, obfuscation

17 Manufacturing cheap, resilient, and stealthy opaque constructs
Christian Collberg, Clark Thomborson, Douglas Low
January 1998 **Proceedings of the 25th ACM SIGPLAN-SIGACT symposium on Principles of programming languages POPL '98**
Publisher: ACM Press
Full text available: pdf(1.59 MB) Additional Information: full citation, references, citings, index terms

18 Privacy preservation and social issues: A privacy-preserving interdomain audit framework
Adam J. Lee, Parisa Tabriz, Nikita Borisov
October 2006 **Proceedings of the 5th ACM workshop on Privacy in electronic society WPES '06**
Publisher: ACM Press
Full text available: pdf(4.55 MB) Additional Information: full citation, abstract, references, index terms

Recent trends in Internet computing have led to the popularization of many forms of virtual organizations. Examples include supply chain management, grid computing, and collaborative research environments like PlanetLab. Unfortunately, when it comes to the security analysis of these systems, the whole is certainly greater than the sum of its parts. That is, local intrusion detection and audit practices are insufficient for detecting distributed attacks such as coordinated network reconnaissance, ...

Keywords: data obfuscation, distributed audit, logging

19 Session 2: Review and analysis of synthetic diversity for breaking monocultures
James E. Just, Mark Cornwell
October 2004 **Proceedings of the 2004 ACM workshop on Rapid malcode WORM '04**
Publisher: ACM Press
Full text available: pdf(356.14 KB) Additional Information: full citation, abstract, references, citings, index terms

The increasing monoculture in operating systems and key applications and the enormous

expense of N-version programming for custom applications mean that lack of diversity is a fundamental barrier to achieving survivability even for high value systems that can afford hot spares. This monoculture makes flash worms possible. Our analysis of vulnerabilities and exploits identifies key assumptions required to develop successful attacks. We review the literature on synthetic diversity techniques, f ...

Keywords: diversity, n-version programming, vulnerability

20 An abstract interpretation-based framework for software watermarking
Patrick Cousot, Radhia Cousot
January 2004 **ACM SIGPLAN Notices , Proceedings of the 31st ACM SIGPLAN-SIGACT symposium on Principles of programming languages POPL '04**, Volume 39 Issue 1
Publisher: ACM Press
Full text available: pdf(171.12 KB) Additional Information: full citation, abstract, references, citings, index terms

Software watermarking consists in the intentional embedding of indelible stegosignatures or watermarks into the subject software and extraction of the stegosignatures embedded in the stegoprograms for purposes such as intellectual property protection. We introduce the novel concept of *abstract software watermarking*. The basic idea is that the watermark is hidden in the program code in such a way that it can only be extracted by an abstract interpretation of the (maybe non-standard) concre ...

Keywords: abstract interpretation, authentication, copyrights protection, fingerprinting, identification, intellectual property protection, obfuscation, software authorship, software watermarking, static analysis, steganography, stegoanalyst, stegoattacks, stegokey, stegomark, stegosignature, tamper-proofing, trustworthiness, validation watermarking

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

PORTAL

USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: The ACM Digital Library The Guide

SEARCH

Feedback Report a problem Satisfaction s

Protecting Java code via code obfuscation

Full text Html (37 KB)

Source **Crossroads** archive
Volume 4 , Issue 3 (Spring 1998) table of contents
Speical issue on robotics
Pages: 21 - 23
Year of Publication: 1998

Author Douglas Low Univ. of Auckland, Auckland, New Zealand

Publisher ACM Press New York, NY, USA

Additional Information: abstract references cited by index terms collaborative colleagues

Tools and Actions: Find similar Articles Review this Article
Save this Article to a Binder Display Formats: BibTex EndNote ACM Ref

DOI Bookmark: Use this link to bookmark this Article: http://doi.acm.org/10.1145/332084.332092
What is a DOI?

↑ ABSTRACT

The Java language is compiled into a platform independent bytecode format. Much of the information ‹
the original source code remains in the bytecode, thus decompilation is easy. We will examine how coc
obfuscation can help protect Java bytecodes.

↑ REFERENCES

Note: OCR errors may be found in this Reference List extracted from the full text article. ACM has opte
expose the complete List rather than only correct and linked references.

1 Alfred V. Aho , Ravi Sethi , Jeffrey D. Ullman, Compilers: principles, techniques, and tools, A‹
Wesley Longman Publishing Co., Inc., Boston, MA, 1986

2 David F. Bacon , Susan L. Graham , Oliver J. Sharp, Compiler transformations for high-perfo‹
computing, ACM Computing Surveys (CSUR), v.26 n.4, p.345-420, Dec. 1994

3 3 Bank, Joseph A. Java Security. http://swissnet.ai.mit.edu/~jbank/javapaper/javapaper.htn
December, 1995.

4 Cristina Cifuentes , K. John Gough, Decompilation of binary programs, Software—Practice &
Experience, v.25 n.7, p.811-829, July 1995

5 5 Collberg, Christian, Clark Thomborson and Douglas Low. A taxonomy of obfuscating transf‹
Technical Report 148, Department of Computer Science, University of Auckland, New Zealan‹
1997.
http://www.cs.auckland.ac.nz/~collberg/Research/Publications/CollbergThomborsonLow97a/

6 Christian Collberg , Clark Thomborson , Douglas Low, Manufacturing cheap, resilient, and ste opaque constructs, Proceedings of the 25th ACM SIGPLAN-SIGACT symposium on Principles programming languages, p.184-196, January 19-21, 1998, San Diego, California, United Sta

7 James Gosling , Bill Joy , Guy L. Steele, The Java Language Specification, Addison-Wesley Lo Publishing Co., Inc., Boston, MA, 1996

8 Amir Herzberg , Shlomit S. Pinter, Public protection of software, ACM Transactions on Compu Systems (TOCS), v.5 n.4, p.371-393, Nov. 1987

9 9 Jaeschke, Rex. Encrypting C source for distribution. *Journal of C Language Translation*, vol. 1990. http://jclt.iecc.com

10 10 Jokipii, Eron. Jobe - The Java obfuscator. http://www.primenet.com/~ej/index.html, 199

11 11 LaDue, Mark D. HoseMocha. http://www.oasis.leo.org/java/development/bytecode/obfuscators/HoseMocha.dsc.html

12 Tim Lindholm , Frank Yellin, Java Virtual Machine Specification, Addison-Wesley Longman Pu Co., Inc., Boston, MA, 1999

13 John J. Marciniak, Encyclopedia of software engineering, Wiley-Interscience, New York, NY, 1

14 14 Sape Mullender, editor. *Distributed Systems*. Addison-Wesley, 2nd edition, 1993.

15 R. L. Rivest , A. Shamir , L. Adleman, A method for obtaining digital signatures and public-ke cryptosystems, Communications of the ACM, v.21 n.2, p.120-126, Feb. 1978

16 16 van Vliet, Hans Peter. Crema - The Java obfuscator. http://java.cern.ch/Java/java/CremaE1/DOC/Index.html, January, 1996.

17 17 van Vliet, Hans Peter. Mocha - The Java decompiler. http://java.cern.ch/Java/java/MochaB1/Readme.txt, January, 1996.

18 18 Wilhelm, Uwe G. Cryptographically protected objects. http://lsewww.epfl.ch/~wilhelm/Cr May, 1997. A french version appeared in the Proceedings of RenPar'9, Lausanne.

19 19 WingSoft Company. JavaDis - The Java Decompiler. http://www.wingsoft.com/wingdis.sh 1997.

↑ CITED BY *4*

Ginger Myles, Using software watermarking to discourage piracy, Crossroads, v.10 n.3, p.2-2, 31 2004

Ginger Myles, Using software watermarking to discourage piracy, Crossroads, v.12 n.1, p.4-4, Fa

Kung Chen , Ju-Bing Chen, On instrumenting obfuscated java bytecode with aspects, Proceedings 2006 international workshop on Software engineering for secure systems, May 20-21, 2006, Sha China

Jien-Tsai Chan , Wuu Yang, Advanced obfuscation techniques for Java bytecode, Journal of Syste Software, v.71 n.1-2, p.1-10, April 2004

↑ INDEX TERMS

Primary Classification:

D. Software

↪ **D.3** PROGRAMMING LANGUAGES

↪ **D.3.2** Language Classifications

↪ **Nouns:** Java

Additional Classification:
D. Software
↪ **D.3** PROGRAMMING LANGUAGES

General Terms:
Design, Languages, Performance, Theory

↑ **Collaborative Colleagues:**
Douglas Low: Christian Collberg
Clark Thomborson

Refine Search

Search Results -

Terms	Documents
L16 and (instruction ADJ counter)	15

Database:
US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Search: L17

Refine Search

Recall Text Clear Interrupt

Search History

DATE: Saturday, May 12, 2007 Purge Queries Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=PGPB; PLUR=NO; OP=OR			
L17	L16 and (instruction ADJ counter)	15	L17
L16	L15 and (program ADJ counter)	187	L16
L15	717/141.ccls. or 713/190,193,194.ccls. or 714/28,30.ccls. or 712/226,227.ccls.	1817	L15
L14	L13	0	L14
DB=USPT; PLUR=NO; OP=OR			
L13	L12 and (program ADJ counter)	12	L13
L12	L11 and (instruction ADJ counter)	38	L12
L11	717/141.ccls. or 713/190,193,194.ccls. or 714/28,30.ccls. or 712/226,227.ccls.	3170	L11
L10	L8 NOT L9	89	L10
L9	L8 and (obfuscate or obfuscated or obfuscation).ab.	3	L9
L8	L7 or L6 or L5	92	L8

L7	L2 and (program adj counter)	91	L7
L6	L2 and (instruction adj counter)	1	L6
L5	L3 and (program adj counter)	12	L5
L4	L3 and (instruction adj counter)	0	L4
L3	L2 and ((smart ADJ card) or (java ADJ card))	59	L3
L2	obfuscate or obfuscated or obfuscation	1083	L2
L1	obsfuscation	0	L1

END OF SEARCH HISTORY